INTRODUCTION A MATLAB

***AVANT-PROPOS** :*
***Cette introduction s'adresse à des personnes** possédant des notions d'Algèbre (vecteurs et matrices), **mais débutantes complètes en calcul scientifique sur ordinateur**.*
Dans le maniement d'un outil informatique, le plus difficile, c'est le début du début du commencement, n'est-il pas vrai ? C'est pénétrée de cette vérité profonde que j'ai rédigé, pour quelqu'un que je ne connaissais pas et qui demeurait à plusieurs milliers de kilomètres, cette brève introduction à Matlab. J'ai voulu avant tout éviter que ce débutant se trouve bloqué par une de ces difficultés que l'on trouve à la fois idiotes et inavouables ... quand on les a résolues. Aussi ai-je cherché la plus grande simplicité ; peu m'importe de faire rire le praticien expérimenté, si je parviens à aider le néophyte.

***CONSEILS A L'UTILISATEUR** :*
"MATLAB" est l'abrégé de "Matrix laboratory", ce qui signifie que cet outil a été conçu pour manipuler vite et bien les tableaux de nombres, notamment les matrices. D'où :
- si nécessaire, révisez, en Algèbre, les opérations portant sur les vecteurs et les matrices ;
- en Matlab, les manipulations de tableaux se font de façon globale. Conséquence importante :
***Moins il y a de boucles, plus le calcul est rapide**.*

Matlab cours n°1

Préalable : l'ordinateur est allumé, le programme Matlab est installé.

Double-cliquer sur l'icône Matlab pour commencer une session Matlab.

Nous allons travailler aujourd'hui uniquement à la fenêtre de commande, fenêtre surmontée par la mention
 "command window".

Nous allons exécuter une succession de commandes Matlab. Chaque commande doit être écrite après le signe >> qui s'inscrit automatiquement à la fenêtre de commande.

Après avoir terminé l'écriture d'une commande, pour déclencher son exécution, il faut appuyer sur la touche "Entrée" (comme on fait pour aller à la ligne quand on écrit un texte).

Dans ce cours, j'écris les commandes Matlab en gras pour les distinguer du reste du texte, mais sous Matlab on ne les écrit pas en gras mais dans le style standard. Voici une première série de

commandes à exécuter :

```
a = 1
b=a + 2
pi
L=[-1 2 3]
C= [8 ; 2 ; 4]
 LT=L'
S=C + LT
SP1 = S + 1
Ppelem = LT .* C
Rpelem = LT ./ C
proscal = L * C
prodya = C * L
```

Note : bien étudier la différence entre * tout seul et * précédé d'un point. Essayer avec d'autres exemples.
Les lettres majuscules et les minuscules sont distinguées les unes des autres. Les variables que l'on a définies demeurent présentes dans la mémoire de l'ordinateur, utilisables pour d'autres calculs. On obtient un panorama général des variables (scalaires et tableaux) définies depuis l'ouverture de la session à l'aide de la commande :

```
whos
```

Pour connaître la taille d'un seul tableau, on utilise la commande "size". Exemple :

```
size(prodya)
```

Pour localiser un élément d'un tableau (afin de connaître ou de modifier sa valeur), on précise son rang dans le tableau. Exemples :

```
prodya(2,2)
L(3)
L(3) = L(2)
L
```

Graphique :
Les tableaux unidimensionnels (vecteus lignes ou colonnes) peuvent être représentés à l'aide de la commande "plot", les tableaux bidimensionnels (matrices) à l'aide de la commande "surf ".
Exemples :

```
plot(L)
plot(C,'*')
surf (prodya)
```

Pour plus de renseignements sur la commande plot(en particulier les diverses sortes de traits), utiliser la commande :

help plot

La prochaine fois, nous étudierons les tableaux de dimension deux.

MATLAB cours n°2

Rappel : dans le cours précédent, nous avons vu la définition d'un scalaire et celle d'un petit vecteur ligne ou colonne, ainsi que diverses opérations (+, *, .*, ./).

Aujourd'hui, nous étudions (définition et opérations) les petites matrices, puis nous passons à des vecteurs et matrices plus importants.

1) Petites matrices :

Même règle d'écriture que pour les vecteurs : le point-virgule est le séparateur de lignes ; à l'intérieur d'une ligne, l'espace est le séparateur de colonnes
Commandes Matlab :

```
A= [1 0 2; 5 -2 8 ]
B= A-2
C=B*10
D=(A-2)*10
rien = C - D
PPE=A.*B
AT = A'
PCAR  = A * AT
GCAR = AT * A
CARelem = A.*A
CC=A.^2
RAC = sqrt(CC)
```

Encore une fois, **bien examiner la différence entre les opérations * (produit de matrices) et .* (produit élément par élément)**. Le produit * (étoile toute seule) s'utilise selon les règles de l'algèbre, avec les contraintes habituelles sur les dimensions des tableaux . Ses résultats sont, toujours selon la règle algébrique du produit des matrices :

matrice * vecteur colonne = vecteur colonne
vecteur ligne * vecteur colonne = scalaire
vecteur colonne * vecteur ligne = matrice
matrice * matrice = matrice

Le produit .* s'utilise pour deux tableaux de même dimension ; le résultat est un tableau de même dimension que ceux de départ. L'élément n° 3 de A.*B est le produit du troisième élément de A

par le troisième élément de B. Il existe de même la division élément par élément notée ./

Opérations faisant intervenir des scalaires :
A étant un tableau (vecteur ligne, vecteur colonne, matrice), B = A + 2 signifie que chaque élément du tableau B est obtenu en ajoutant deux à l'élément correspondant du tableau A. La différence fonctionne de façon analogue. On peut diviser ou multiplier tout un tableau par un scalaire sans avoir à écrire le point devant la barre oblique ou l'étoile.

<u>Sélection d'une partie de matrice</u> :
- A(1,2) est l'élément première ligne, deuxième colonne de A ;
- A(:,1) est le vecteur colonne formé par la première colonne de A
- A(2,:) est le vecteur ligne formé par la deuxième ligne de A.

<u>Exercice : résolution d'un système linéaire</u> :
Considérons le système suivant, où x, y, z sont inconnus :
2 x + 3 z = - 2
x - 2 y + 8 z = - 3
6 x + z = - 6
Ce système peut s'écrire sous la forme :
A * V = W (1)
avec :
A matrice 3 x 3, W vecteur colonne formé par les seconds membres des trois égalités ci-dessus, V vecteur colonne inconnu formé par les trois inconnues x, y, z.

1) sur papier, écrire le système sous la forme matrice *vecteur colonne = vecteur colonne en détaillant tous les coefficients, connus (nombres) ou inconnus (x, y, z).
2) sous Matlab, définir par les commandes convenables la matrice A et le vecteur colonne W.
3) utiliser la commande Matlab suivante qui donne la solution (lorsqu'elle existe) du système linéaire :

V = A \ W
(attention, ce n'est pas / mais bien \ ; pour des détails : help slash))

4) vérifier l'exactitude de la solution obtenue en faisant :

W1 = A * V

5) on aurait pu faire aussi

V = inv(A) * W
Mais le temps de calcul est plus grand

La prochaine fois : grands tableaux, courbes

<u>MATLAB cours n°3</u>

Rappel : dans les cours précédent, nous avons vu comment définir de petits tableaux et faire sur eux diverses opérations algébriques.

Aujourd'hui, nous étudions comment définir et représenter des tableaux plus grands.

- Grands vecteurs:

Pour définir un tableau X de nombre allant de -4 à +5 par pas de 0.1, on utilise la commande :

x = -4 : 0.1 5

Pour définir le même tableau, mais sans faire apparaître à l'écran la liste des éléments, on termine la ligne de commande par un point virgule :

x = -4 : 0.1 5 ;

essayer aussi :

y = -8 : 3

yy = (-5 : 5) * 0.1

u = 5 : - 1 : - 3

application : illustration de y = f(x)

x = -10 : 0.05 : 10 ;

plot(x*180/pi, sin(x),'r')

xlabel('angle x en degrés')

title('illustration de sin(x)')

Pour en savoir plus sur la commande graphique "plot" :

help plot

pour voir la liste des nombreuses fonctions mathématiques disponibles :

help elfun

help specfun

2) Grandes matrices

(les commandes suivantes permettent d'engendrer des tableaux de grande dimension, même si je propose des exemples en dimension modeste pour une vue claire du résultat à l'écran)

A=zeros(5,3)

B=ones(4,6)

C=eye(5,5)

R=randn(20,25);

surf(R)

UN=ones(5,4)

LARGE = [UN A UN] %concaténation de trois matrices

Notes : ce qui est à droite de "%" est considéré comme un commentaire ; la concaténation se pratique aussi pour les vecteurs.

2) Fonction de deux variables

On a souvent besoin de calculer une fonction f(x,y) sur un maillage de points du plan (x,y). Pour définir ce maillage, il existe la commande meshgrid :

x = 0 : 5

y = -1 : 2

[X,Y] = meshgrid(x,y)

Z = X.^2 +Y.^2

surf(Z)

Note : la figure est plus belle avec des tableaux plus grands !

Les deux tableaux bidimensionnels X et Y obtenus à l'aide de meshgrid auraient pu être engendrés par les opérations algébriques suivante :
XX=ones(length(y),1)*x
YY=y'*ones(1,length(x))

4) Rappels sur les dimensions de tableaux :
pour un panorama général :
whos
pour connaître la dimension du tableau (vecteur ou matrice) A :
size(A)
pour connaître le nombre de coefficients du vecteur V :
length(V)

5) Impression ou copie
On peut sélectionner avec la souris une partie de ce qui est écrit à la fenêtre de commande. Cette sélection peut ensuite être imprimée (menu File) ou copiée dans le presse-papier (menu Edit). Une figure possède elle aussi des menus File et Edit qui en permettent l'impression ou la copie.

La prochaine fois, nous verrons comment créer et exécuter des fichiers de programmes et de fonctions.

MATLAB cours n°4

Dans les cours précédents, nous avons travaillé à la fenêtre de commande. Aujourd'hui, nous écrivons un programme (ou logiciel). Un programme est un fichier contenant une succession de commandes. Quand on demande l'exécution du programme, ces commandes sont exécutées les unes après les autres, dans l'ordre où elles figurent dans le programme. Un programme est une archive qui n'est pas effacée par la fermeture de la session Matlab ni par le fait d'éteindre l'ordinateur.

Ecriture du programme:
En haut à gauche de la fenêtre de commande, activer le menu File, puis New, puis M-File.Ceci fait apparaître une nouvelle fenêtre dans laquelle vous allez écrire la liste des commandes constituant le programme (ici, il n'y a pas de >>) :

```
% premier.m : mon premier programme Matlab
x = -10 : 0.1 : 10
 y=sin(0.2*x)+sin(x)+sin(1.5*x);
figure(1),  plot(x,y,'r')
title('illustration de y=sin(0.2*x)+sin(x)+sin(1.5*x)')
figure(2), plot(x, exp(-0.1*x.^2), title('gaussienne'), grid
%fin du programme
```

En haut de la fenêtre “untitled” où vous avez écrit le programme, activer le menu File puis Save As. Il vous est alors proposé de sauvegarder le programme dans l’espace de travail par défaut qui est MATLAB6p1\work ; vous donnez un nom (par exemple “premier”) et vous cliquez sur le bouton “enregistrer”. Pour exécuter le programme : à la fenêtre de commande, tapper le nom du programme sans l’extension .m. :
premier

Gestion de la mémoire
Quand on a exécuté un programme, tous les résultats des définitions et des calculs sont disponibles dans la mémoire de l’ordinateur. Pour s’en assurer :
whos
Pour les supprimer si nécessaire :
clear all

Imbrications :
Un programme peut contenir, entre autres commandes, le nom d’un autre programme. La rencontre de cette commande particulière déclenche l’exécution du programme ainsi appelé. Une fois cette exécution terminée, on continue l’exécution de la suite des commandes du programme appelant.

Eléments de programmation
On trouve en Matlab comme dans les autres langages de programmation les instructions “for”, “while”, “if”. Attention : **Matlab n’est pas rapide pour l’exécution de ces commandes**. Si le calcul que l’on veut faire nécessite vraiment un grand nombre de ces instructions, il vaudra mieux l’écrire dans un autre langage. Mais beaucoup de boucles sont évitables sous Matlab.

Exemple de boucle évitable :
multiplier par 10 tous les éléments d’un tableau A. Il ne faut surtout pas faire une boucle dans laquelle on écrirait par exemple B(n) = A(n) * 10, en faisant varier n de 1 au nombre d’éléments du tableau A. . Il suffit d’écrire (l’ exécution est beaucoup plus rapide) :
B = A * 10 ;

Exemple de boucle inévitable :
Calcul des termes successifs d’une suite définie par récurrence.

Fonctions Matlab
Il est parfois plus commode d’appeler une fonction qu’un programme. Attention : sous Matlab, concernant les fonctions, il y a diverses obligations à respecter..
1) Une fonction doit avoir comme première ligne de commande (non précédée par un commentaire %) :

```
function nomduresultat = nomdelafonction(nomdelargument)
```

2) Pour compatibilité avec les versions Matlab datant de quelques années, écrire toujours une fonction seule dans un fichier qui porte son nom.
exemple :

```
function resu = jondule(x)
resu = sin(x) + sin(2*x) + sin(3*x);
%fin de jondule
```

Cette fonction sera stockée dans le fichier jondule.m. Séquence d'appel dans le programme principal :

```
x = -5 : 0.02 : 5;
y = jondule(x) ;
plot(x, y)
```

Nous avons vu qu'après exécution d'un programme tous les résultats étaient disponibles dans la mémoire de l'ordinateur. Après exécution d'une fonction, il n'en va pas de même : seul est disponible le résultat indiqué dans la première ligne de la fonction. Je conseille au débutant de se familiariser d'abord avec les programmes avant de se mettre aux fonctions.

La prochaine fois (qui sera la dernière de cette introduction), nous verrons les nombres complexes et quelques compléments sur les graphiques.

MATLAB cours n°5

Lors de cette séance, qui est la dernière dans la rubrique "Introduction à Matlab", nous étudions les calculs en nombres complexes et quelques compléments graphiques.

1 - Nombres complexes :

A l'ouverture de la session Matlab, les variables i et j sont prédéfinies :

```
M1 = i ^2
m1 = j ^2
sqrt(-1)
```

Attention : si l'on donne la commande i = 3, désormais le carré de i est égal à 9. Le plus simple pour réparer l'erreur est de fermer la session.

Définition genre z = x + i y :

```
x= randn(1,10)
y = randn(1,10)
z = x + i * y
transpconj = z'
```

Essayer ensuite les commandes real(z), imag(z), abs(z), angle(z). La plupart des fonctions mathématiques disponibles sous Matlab acceptent des arguments complexes (essayer sqrt, sin, log...)

Définition genre z = r $e^{i\theta}$

```
teta = (0:5:360)*pi/180
Z = exp(i*teta) ;
```

figure(1) , plot(real(Z),imag(Z))
spirale=teta.*Z;
figure(2), plot(real(spirale),imag(spirale))

Dessin :
Si Z est un tableau unidimensionnel de nombres complexes, plot(Z) dessine la partie réelle en abscisse et la partie imaginaire en ordonnée. X étant un tableau réel de même dimension que Z, plot(X,Z) dessine la partie réelle de Z en fonction de X .

2 – Complément sur les graphiques
Nous avons vu comment ouvrir deux (ou plus) fenêtres graphiques. On peut aussi superposer une nouvelle courbe sur un graphique déjà tracé :
plot(teta, real(Z),'k')
hold on
plot(teta, imag(Z), 'r')

Pour arrêter les superpositions, utiliser "hold off"
On peut sélectionner une partie d'un graphique, ou modifier les échelles, à l'aide de la commande "axis" (voir l'aide par "help axis"). Comme hold on, axis, et aussi title, xlabel, ylabel … opèrent sur des figures déjà tracées (c'est à dire que la commande "plot" doit précéder ces commandes).

Une commande graphique est exécutée sur la dernière fenêtre graphique que l'on a nommée (par "figure(n)") ou sur laquelle on a cliqué.

Copie dans Word : pour éviter d'avoir des figures avec un fond noir :dans Matlab, fenêtre graphique, menu file, rubrique préférences > figure copy template, appuyer sur le bouton "word".

3 – Pour en savoir plus de façon générale
Nous avons déjà vu des exemples d'utilisation de l'aide en ligne :
help axis
L'aide obtenue de cette façon concerne une commande précise. Pour obtenir une aide plus générale, genre manuel d'utilisation, utiliser le menu "help" en haut de la fenêtre de commande.
Regarder les exemples (rrubrique "Demos" dans le menu help) est une bonne façon de s'instruire sans se fatiguer.
Quand une "demo" vous intéresse particulièrement, on peut faire apparaître la liste du programme :
type sphere.m

3 – Ce que je ne traite pas ici pour le moment :

3) Il existe des fonctions Matlab (intégration, moindres carrés…) utilisant comme argument une fonction écrite par vous.
4) On peut illustrer de façon sonore des tableaux de nombres :
   help sound
5) Il existe de très nombreux ordres graphiques permettant en particulier des animations (voir les démos)

6) Il existe la possibilité de ne pas payer (de façon tout à fait légale), en utilisant au lieu de Matlab le logiciel Scilab (diffusé par l'INRIA, téléchargeable) qui est gratuit. Tous les calculs algébriques se font de la même façon qu'avec Matlab. Par contre, les ordres graphiques diffèrent.

C'est tout pour l'instant. A suivre...

Françoise Depasse

contact : lettre.francoise@yahoo.fr
adresse du site : http://fd230209.site.voila.fr/
adresse de cette page : http://fd230209.site.voila.fr/Matlab_Intro.pdf